AFRL-OSR-VA-TR-2014-0167

INFORMATION HIDING BASED TRUSTED COMPUTING SYSTEM DESIGN

Gang Qu
MARYLAND UNIV COLLEGE PARK

07/18/2014
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ RTC
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 30-06-2014	**2. REPORT TYPE** Final Report	**3. DATES COVERED** *(From - To)* 1, May 2010 - 31, March 2014

4. TITLE AND SUBTITLE Information Hiding based Trusted Computing System Design	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER FA9550-10-0140
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Qu, Gang; Wu, Min	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Maryland, Office of Research Administrtaion & Advancement, 3112 Lee Building, College Park, MD 20742-5100	**8. PERFORMING ORGANIZATION REPORT NUMBER** 014272
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFOSR/PKR1, AF Office of Scientific Research, 875 North Randolph Street, Room 3112, Arlington, VA 22203	**10. SPONSOR/MONITOR'S ACRONYM(S)** AFOSR
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This proposal is a three-year research project to enhance trust in computing system design, which we have achieved by a combination of research (21 published work), education (4 graduated Ph.D. students and 2 undergradauted courses), and presentation (18 invited talks). We focus on the design and implementation of sequential circuits, a crucial component for any real-time computing systems. More specifically, we demonstrate that (1) severe vulnerabilities exist in systems designed with today's design methodology; (2) an adversary can embed hardware Trojan without compromising design quality or being detected; and (3) practical design techniques can be developed to establish trust in sequential design. We have also investigated how to enhance system's trust by leveraging both the underlying hardware of the system (silicon physical unclonable functions) and the environment where the system operates (electrical network frequency signals), and how to improve the trust in a wireless sensor network with the presence of insider attackers. This project has contributed to raising the global awareness of hardware security and trust. The PI is currently working with AFRL researchers through a summer faculty program to expand trust design to nanoscale devices.

15. SUBJECT TERMS

Hardware security, trusted computing, silicon physically unclonable funcation (PUF), electrical network frequency (ENF), trust in wireless sensor network.

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Qu, Gang
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(Include area code)*
U	U	U	SAR	20	301-405-6703

Final Report for AFOSR Grant#FA9550-10-1-0140

Project Title: **Information Hiding based Trusted Computing System Design**

Investigator team: Dr. Gang Qu (PI) and Dr. Min Wu (Co-PI)
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742

Contact Information: Dr. Gang Qu
gangqu@umd.edu
Tel: 301-405-6703
Fax: 301-314-9281;

Table of Contents

Abstract

This proposal is a three-year research project to ***enhance the trust of computing system design***. We use the term "***design***" for the entire process of delivering a working system from its high level specification, which includes software development, hardware design, as well as chip fabrication and testing. We use the term "***trust***" for the guarantee that the delivered system does exactly what it is asked for, no more and no less, which is particularly critical for defense related systems.

We have accomplished the goal of "enhancing the trust of computing systems" through a combination of research (**21 published work**), education (**4 graduated Ph.D.**), and presentation (**18 invited talks**). As we have planned in the original proposal, we focus on the design and implementation of sequential circuits, a crucial component for any real-time computing system. More specifically, we demonstrate that (1) severe vulnerabilities exist in the systems designed with today's design methodology; (2) an adversary can easily embed hardware Trojan horse without compromising the design quality and without being detected; and (3) practical design techniques can be developed to establish trust in sequential design [1-3]. This part of the research will be elaborated in details in the following sections.

In addition, we have worked on several other topics related to the design of trusted systems. First, we investigate how a newly discovered phenomena, known as silicon physical unclonable functions (PUF), can help to build trust in computing systems. A silicon PUF is a circuitry that can capture the fabrication variation, which is believed to be random and unique for each chip, and use such information for many security related applications such as device authentication and secret key generation. Our work focuses on reducing the hardware cost of PUF [4-7], improving PUF secret's reliability and usability [8-10], and finding new applications for PUF [11,12]. Second, we study the electrical network frequency (ENF) signals from the power grid and discover that the ENF signals also carry unique features that can be used to estimate the geo-location of a device [13-18]. With the ubiquitous of power grid and ENF signals, this opens the door to another novel and powerful tool to improve the trust of computing systems. Finally, we consider wireless sensor networks and demonstrate how to counter insider attacks by proposing new trust metrics and developing novel mitigation methods [19-21].

This project, as an important piece of the collaborative efforts with many other researchers, has already made significant impact. The global awareness of hardware security and trust has reached an unprecedentedly high level as shown by the first DARPA MURI topic on nanoscale devices design security, which is managed by AFOSR program manager who has funded the PI's work and the PI is a member of the winning team; dozens of workshop and conference special sessions on hardware security; and hundreds of research publications. Following the recommendation of the AFOSR program manager, the PI is working with AFRL (Rome, NY) researchers through a summer faculty program to continue this current project and expand its scope to nanoscale devices.

1. Introduction and Motivations

In the past few years, there has been a fast increasing interest in building integrated circuit (IC) based trusted systems [22-25], mainly driven by the prolific applications, both military and civilian, that require security and access control. As EDA and semiconductor continue to evolve rapidly, a user of the system very unlikely will have the expertise and capability to do in-house design and fabrication for the system he wants to build. Once he gives the design specification to a design house for design and later on gives the design information to a foundry for manufacture, the user will lose full control of the system's functionality and specification (spec). It is not sufficient to use only trusted design house and foundry because the data communication among them can never be guaranteed secure. Even with an ideal testing tool that can verify correctly whether all the design specs have been met or not, the system or IC remains untrustworthy until the user is convinced that no additional functionality is intentionally hidden in the system or IC. Therefore, we use the term "***trust***" for the guarantee that the delivered system does exactly what it is asked for, no more and no less, which is particularly critical for defense related systems [26].

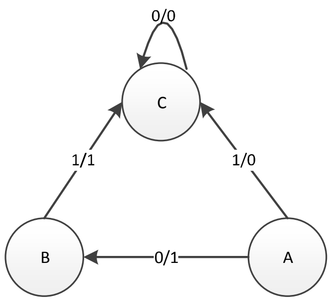
To ensure the user that the system does each of the required functionalities (the "no less" part of trust), the design house only needs to show that expected output will be produced under the given testing input data. However, *verifying that an IC or a system is not intentionally designed to perform any undesired functionality* (the "no more" part of trust) is much harder. Particularly, in most cases the user may not know the undesired functionalities he wants to verify. To verify the non-existence of known unwanted functionality, one can use mathematical techniques such as proof by contradiction or conduct a careful testing. But there is no systematic method to verify the non-existence of unknown functionalities.

This has also been identified in the Defense Science Board study on High Performance Microchip Supply, "*Trust cannot be added to integrated circuits after fabrication; electrical testing and reverse engineering cannot be relied on to detect undesired alternations in military integrated circuits*" [26]. The Department of Defense's (DOD) Trusted Foundry Program and the complementary Trusted IC Supplier Accreditation were specifically designed for domestic fabrication, where trust and accreditation are built on reputation and partnership [27]. With the predicted dramatic drop of number of foundries starting in 2009 [28], there is the urgent need to find solutions for establishing trust in ICs. Similar problems exist in software development and the design of any computing devices with third-party involvement. It is thus desirable to investigate a unified framework to address the trust issues in ensuring a delivered system does no more and no less than what it is asked for.

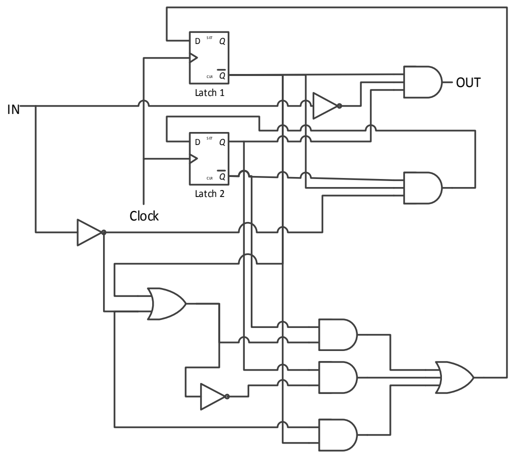
Sequential components are crucial for real time embedded systems as they control the system based on the system's current state and real life input. In the rest of this section, we explore the security and trust issues of sequential system design from the perspective of finite state machine (FSM), which is the most popular model to describe sequential systems. Consider the 3-state finite state machine (FSM) shown in Figure 1(a); we follow the standard procedure to implement this sequential system with two flip flops (FF) and some logic gates (Figure 1 (b)).

First, from Figure 1(a), we see that when the system is at state B and input is 0, no next state and output are specified, which are known as *don't care transitions*. However, in the circuit level implementation when FF1 is 0 and FF2 is 1, which reflects the current state B, if input $x = 0$, we can easily verify that FF1 remains unchanged, but FF2 changes to 0. This means that the system switches to state A. At the same time, we can see that the output will be 1. This corresponds to the dashed line from state 01 to state 00 in Figure 1(c). Similarly, state 10 will move to state 00 and output 0 on input $x = 1$.

a The original 3-state FSM as the system specification. The label on each edge (such as 0/1 from state A to state B) indicates that the transition occurs on input '0' and the transition results in an output '1'.



b The logic/circuit implementation of the 3-state FSM shown in **a**.



c The 4-state FSM generated from the circuit shown in **b**.

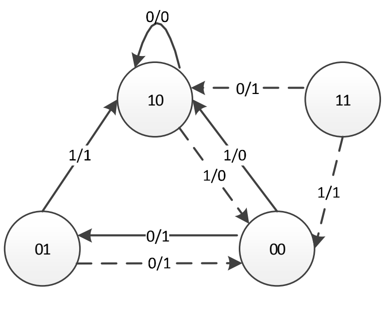


Fig. 1 The illustrative example. States A, B, and C in **a** correspond to the states 00, 01, and 10, respectively in **c**. The dashed edges are the transitions implemented by the circuit in **b** but not required by the FSM in **a**.

Second, when both flip flops have value 1, the system will be in a state that is not specified in the original FSM. With input $x = 0$ and $x = 1$, the system will output 1 and move to state C and state A, respectively. In other words, the full functionality of the circuit in Figure 1(b) can be described as is shown in Figure 1(c).

The FSM in Figure 1(a) is what we want to design, the FSM in Figure 1(c) is the system that the circuit we are given (Figure 1(b)) actually implements. Clearly we can see the difference between these two FSMs. The one in Figure 1(c) has one more state and four more transitions. It is this added state and transitions that creates security and trust concerns for sequential system design.

In the original FSM (Figure 1(a)), when the system leaves state A, it cannot come back. In other words, we say that there is no access to state A from state B and state C. However, in the implemented FSM in Figure 1(c), each state has at least one path to come back to state 00. For instance, from state 01 (which is state B in the original FSM), when we inject 0 as the input, the system moves back to 00 (or state A). If state A is a critical state of the system and we want to control its access, FSM in Figure 1(a) specifies this requirement, but its logic implementation in Figure 1(b) violates this access control to state A (or state 00) and the design cannot be trusted.

This example shows that there are security holes in current sequential system design flow. In the next section, we first lay out a formal model for trust and then report two ways that an attacker can take advantage of these holes to attack untrusted system. Next, we describe a naïve way to build trust, which incur large design overhead and performance degradation. Finally, we elaborate our proposed novel method that can be seamlessly integrated into the current sequential system design flow to establish trust in the design and implementation.

2. Trust, Attacks, and Countermeasure

2.1 Trusted FSM and Trusted Logic Implementation

Intuitively, we can define trust as follows: when a sequential system is specified as an FSM, it is trusted as long as the FSM makes correct transitions from the current state to the next state and produces correct outputs based on the input values. From this definition, it is clear that if an FSM is trusted, all of its equivalent FSMs will be trusted, which implies that whether an FSM is trusted will not change during the state minimization phase. However, this may change during the state encoding and combinational logic design phase of the FSM synthesis.

First, as we have seen from the illustrative example in Figure 1, additional states and additional transitions may be introduced when we design the logic. In the state transition graph, we can have *don't care* conditions where the next state or the output of the transition or both are not specified.

Logic design tools will take advantage of these *don't care* conditions to optimize the design for performance improvement, area reduction, or power efficiency. But when the system (or the FSM) is implemented in the circuit level, these *don't cares* will disappear. The circuit will generate deterministic next states and output for each of the *don't care* conditions. These deterministic values are assigned by CAD tools for optimization purpose and they may make the design untrusted. For example, the next state of a *don't care* transition may be assigned to a state for which access control is required and thus produce an illegal entry to that protected state (i.e., making the state unsafe).

A second type of implicit violation of trust comes from the nature of digital logic implementation. When the original FSM has n states after state minimization, it will need a minimum of $k = \lceil log_2(n) \rceil$ bits to encode these states and some encoding schemes that target other design objectives (such as testability and power) may use even longer codes. As we have seen in the illustrative example, when n is not a power of 2, which happens most of the time, those unused codes will introduce extra states into the system, and all transitions from those extra states will be treated as *don't care* transitions during logic synthesis, introducing uncertainty about the trust of the design and implementation of the FSM.

By analyzing the logic implementation of a given FSM M, we can build an FSM M' that captures the behavior of the circuit. When M and M' are equivalent, we say that the logic implementation is trusted. From the above discussion, we conclude that

Theorem 1. A sequential system will have a trusted logic implementation from the traditional synthesis flow if and only if

a) the system is completely specified
b) the number of states after state reduction is a power of 2
c) code with the minimal length is used for state encoding.

[Proof]: The analysis above shows all the three conditions are necessary. To see they are also sufficient, condition a) indicates that there is no *don't care* transitions; conditions b) and c) guarantee that in the implementation of the system there is no additional states. Therefore, the state transition graph of the sequential system will be unique and cannot be modified. This ensures that the system will be trusted. ■

An ideal trusted IC can be built if the system satisfies the three conditions in Theorem 1. However, it is unrealistic to assume that conditions a) and b) will be satisfied. First, given the complexity of today's system, it is impossible to completely specify the system's behavior of all possible input values. Second, there are no effective methods to ensure that the number of state, after state minimization, will be a power of 2. Finally, without any *don't cares* (condition a)) and no flexibility in choosing the code length (condition c)), the design will be tightly constrained and hard to optimize.

In the experimentation, when we modify the system specification to comply with conditions a)-c), the quality of design drops significantly in terms of area, power, and delay. Detailed experimental results can be found in [1-3].

In the rest of this article, we study the trust of FSMs using state reachability as a metric. Within this context, we consider a given FSM, $M = (V, E)$ (e.g. Figure 1(a)), and its logic implementation (e.g. Figure 1(b)), let $M' = (V', E')$ be the completely specified FSM generated from the logic implementation of M (e.g. Figure 1(c)). Clearly, as graphs, M will be a subgraph of M'. We say that ***the logic implementation of M is trusted*** if for each state $v \in V$ and its corresponding state $v' \in V'$, v and v' have the same reachable sets $R(v) = R(v')$ and the same starting sets $S(v) = S(v')$, where *the reachable set of state* v is the set of all states that the system can reach from v and the *starting set of state* v is the set of states from which the system can reach v. Intuitively, this means that in M', we cannot reach any new states from v ($R(v) = R(v')$) and no new state can reach v either ($S(v) = S(v')$). Apparently, the logic implementation in Figure 1(b) and the corresponding FSM in Figure 1(c) cannot be trusted.

Theorem 2. The following are equivalent definitions for trusted logic implementation: for any state $v \in V$ in an FSM M and its corresponding state $v' \in V'$ in the logic implementation of M,

(1) $R(v) = R(v')\ and\ S(v) = S(v')$
(2) $R(v) = R(v')$
(3) $S(v) = S(v')$

[Proof]: We need to show that (1) $\Leftrightarrow$ (2) $\Leftrightarrow$ (3). Since (1) is the conjunction of (2) and (3), it suffices to show that (2) $\Leftrightarrow$ (3). We prove (2) $\Rightarrow$ (3) by contradiction as follows. (3) $\Rightarrow$ (2) can be proved similarly.

If (2) holds, but (3) does not, then there must exist a pair of states $v \in V$ and its corresponding state $v' \in V'$ such that $R(v) = R(v')$ but $S(v) \neq S(v')$. That is, we can find a state $u \in S(v)$ but its corresponding state $u' \notin S(v')$ or vice versa. From the definition, we know that $u \in S(v)$ is equivalent to $v \in R(u)$. Hence, if we have $u \in S(v)$ but $u' \notin S(v')$ as we just found, we should also have $v \in R(u)$ but $v' \notin R(u')$, which implies that $R(u) \neq R(u')$, contradicting the assumption that (2) holds. ■

2.2 Attacks to Untrusted FSMs

We consider the following two attacking scenarios for the sequential system based on what the adversary can access:

Case I: the adversary can only access the logic implementation of the system or FSM M'. The attacking objective is to gain access to the states that are not accessible as specified in the original specification M. That is, finding paths in M' to states that are unreachable in M.

Case II: the adversary gets hold of the original system specification, M, in the format of FSM and wants to establish a path to reach certain unreachable state without being detected. In this case, the attacker can implement such path into the design. However, the challenge is how to disguise the secret path.

We describe two naive attacks, one for each case. As we will show in the experimental results section, these two simple attacks turn out to be quite powerful and challenging to defend. Therefore, we do not consider any sophisticated attacks although they can be developed.

Attack I: The adversary is aware of the vulnerability of the logic implementation of the FSM following the traditional design flow. Therefore, he can launch the "***random walk attack***" and hope to gain access to states that he is not supposed to reach. In this attack, the adversary will try random input sequences. If it leads to the discovery of a previously safe state (i.e., states that cannot be reached by the adversary according to the design specification), the attack will be successful. This is possible because the FSM synthesis tools will assign values to the *don't care* transitions in order to optimize design objectives such as delay, area, or power. These added transitions may make some of the safe states reachable from states that do not belong to their starting states set and therefore, making them unsafe.

Attack II: In this case, the adversary has the original FSM specification of the system before it is synthesized. If he wants to access state v from a state $u \notin S(v)$, the adversary can simply do the following:

- Check whether there is any *don't care* transition from u, if so, he simply makes v as the next state for that transition. This will give him an unauthorized access to state v in the logic implementation of the system.
- If the state transitions from state u are all specified, he can check whether there are any *don't care* transitions from a vertex/state that belongs to $R(u)$, and try to connect that state to v to create a path from u to v.
- If this also fails, then state v in the system is safe with respect to state u in the sense that one can never reach state v from state u. In this case, the attack fails.

Finally, we mention that in case II, the adversary can take advantage of the new states that logic synthesis tools will introduce (that is, when the number of states is not a power of 2 or non-minimal length encoding is used). He can simply launch attack by connecting any of the new states to state v to gain unauthorized access to state v.

2.3 A Naïve Attempt to Build Trusted FSM

The sufficient and necessary conditions in Theorem 1 for a sequential system to be trustworthy actually gives the following constructive method to build trusted FSM:

i. perform state reduction to reduce the number of states
ii. add new states $\{s1, s2, \dots, sk\}$ to the FSM such that the total number of states becomes a power of 2
iii. add state transitions $\{s1 \rightarrow s2, s2 \rightarrow s3, \dots, sk \rightarrow s1\}$ on any input value.
iv. for the other *don't care* transitions, make s1 as their next state
v. use minimal length codes for state encoding

Apparently, the logic implementation of the FSM following the above procedure satisfies conditions a)-c) in Theorem 1: step iv ensures that the FSM is completely specified; step ii ensures that the number of states is a power of 2; and step v requires the minimal length encoding.

The only non-trivial part of this procedure is the cycle created in step iii. By doing this, we make these new states not equivalent to each other, and thus prevent the FSM synthesis tools from merging these states. This will ensure that the total number of states is a power of 2.

From the analysis in early part of this section, we know that the FSM built by the above procedure will guarantee a trusted logic implementation of the sequential system. However, such implementation will have very high design overhead in terms of area, power and clock speed. The details of this finding is reported in the experimental results section in [3].

2.4 A Practical Approach to Building A Trusted FSM

To reduce the high design overhead for building trusted FSMs, we propose a novel method that combines modification of gate level circuit and the concept of FSM re-engineering introduced in [29]. Before describing our approach, we mention that to limit the access to a protected state v, we need to protect all the states in v's starting set of states. Therefore, in the following discussion, when we consider a state to be protected, we consider all the states in its starting set of states as well.

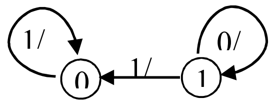


Fig. 2 A simple 2-state FSM.

To illustrate the key idea of our approach, we consider a simple 2-state FSM shown in Figure 2. We assume that state 1 is the safe state and it cannot be reached from state 0. We can add a transition to enforce that the system remains in state 0 on input 0. However, for large design, adding many such transitions will incur high overhead. Instead, we consider how to make state 1 safe at the circuit level. Without loss of generality, we assume that one T flip-flop is used to implement this system. We will use the flip flop content as a feedback signal to control the flip flop input signal (shown as the line with arrowhead in Figure 3). With the help of this new T flip flop, we see that when the system is at state 1, the feedback signal will not impact the functionality of the flip flop input signal. However, when the system is at the normal state 0, the controlled input signal will disable the T flip flop, preventing the system to go to the safe state 1.

Based on this observation, we propose to make the protected states safe by grouping them and allocating them codes with the same prefix (that is, the leading bits of the codes). For example, when the minimal code length is 5 and there are 4 states to be protected, we can reserve the 4 code words 111XX for these 4 states. Then we use flip flops with controlled signals to implement these prefix (like Figure 3 shows). The normal states (i.e., states that we do not want to control their access) will have different prefix and thus any attempt of going to a protected state from the normal states will be disabled.

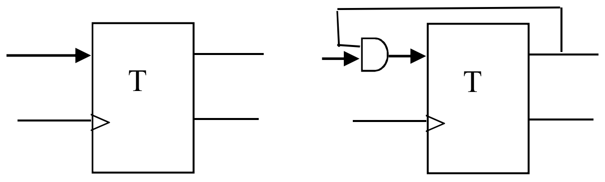


Fig. 3 A normal T flip flop (on the left) and a T flip flop with controlled input (on the right).

However, when the number of states to be protected is not a power of 2, there will be unused codes with the prefix reserved for safe states. If the synthesis tools assign any of these unused code to other states, these states may gain unauthorized access to the protected states and make them unsafe. To prevent this, we apply the state duplication method proposed in [29] to introduce new states that are functionally equivalent to the safe states and mark them also as states to be protected. We repeat this until the total number of safe state becomes a power of 2.

2.5 Summary of Experimental Results

To validate the findings on the security risks of current sequential system design flow and the effectiveness of our proposed approach, a selection of Microelectronics Center of North Carolina (MCNC) sequential benchmark circuits, shown in Table I , in their kiss2 finite state machine format, were used. The first column gives the index for each benchmark circuit. The next 4 columns show, for each circuit, the name, the number of states, the number of input bits, and the number of transitions specified in the blif file [30]. Note that in the blif format, one transition may include multiple input values that move the current state to the next state. For example, if there is a transition from state u

to state v on any of the following input values: 1000, 1001, 1010, and 1011, this will be represented in blif format by only one transition with input 10xx.

The last three columns in the table provide information for a more accurate description of each circuit. We first split each transition into edges, where each edge corresponds to only one input value. For example, the above transition on input 10xx will be split into 4 edges. The column "Number of Edges" gives the total number of edges in each circuit. From this, we can easily calculate the "Number of *don't care* edges" shown in the next column. For example, in the circuit 1 (bbara), there are 4 input bits, which means a total of 2^4=16 different input values. For each of the 10 states, there will be 16 edges, giving a total of 160 edges. The benchmark has 155 edges. So the number of *don't care* edges is 160-155 = 5. The last column gives the number of *don't care* states which can be computed as follows on the same example. We need 4 bits to encode 10 states, but 4 bits will implement 16 states, so the number of *don't care* states is 16 -10 = 6.

Table I MCNC Benchmark Circuit Information

MCNC Circuit Index	Circuit Name	Number of States	Number of Input Bits	Number of Transitions	Number of Edges	Number of Don't Cares Edges	Number of *Don't Care* States
1	bbara	10	4	56	155	5	6
2	bbsse	16	7	53	1800	248	0
3	bbtas	6	2	20	20	4	2
4	beecount	7	3	27	50	6	1
5	dk14	7	3	47	47	9	1
6	dk15	4	3	27	27	5	0
7	dk16	27	2	105	105	3	5
8	dk27	7	1	12	12	2	1
9	dk512	15	1	27	27	3	1
10	ex3	10	2	33	33	7	6
11	ex4	14	6	20	416	480	2
12	ex5	9	2	30	30	6	7
13	ex6	8	5	32	216	40	0
14	ex7	10	2	35	35	5	6
15	keyb	19	7	167	2408	24	13
16	planet	48	7	114	6016	128	16
17	S1488	48	8	250	12160	128	16
18	S1494	48	8	249	12160	128	16
19	s208	18	11	150	36352	512	14
20	sand	32	11	183	63552	1984	0
21	sse	16	7	55	1824	224	0
22	styr	30	9	164	15296	64	2
23	train11	11	2	24	24	20	5
24	train4	4	2	12	12	4	0

In most of these FSM benchmarks, each state is reachable from every other state in the FSM. There is no need to protect such states. To produce FSMs with states to be protected, we modify these benchmarks slightly by removing a small amount of transitions from the blif file so that not all of the states are reachable by all other states. In order to edit the FSMs, a program was written to read in an FSM, then remove transitions, one at a time, and record how many states have become unreachable from other states. This process is repeatable and strictly controlled to prevent an excessive number of transitions from being removed so the modified circuit can still reflect the original benchmark.

The first objective of this work is to demonstrate the vulnerability of traditional FSM synthesis and design flow. We treat states that are not reachable by some states in the FSM as states to be protected and consider all other states as normal states. We then use ABC (a public logic synthesis and formal verification tool) to synthesize each of the FSMs to obtain their circuit implementation. Next we analyze these circuit implementations to generate the completely specified FSM. If the protected states now become reachable from any normal states, these protected states will be considered unsafe. Note that only for four benchmarks (circuits 4, 7, 15, and 19), there are no unsafe states, which means that the circuit implementation of these FSMs are trusted. However, the circuit implementations of the rest of the 20 FSMs are all untrusted.

Our next goal is to show that, given the vulnerability of the FSM synthesis, how attackers can gain unauthorized access to those unsafe states. For this purpose, we consider the aforementioned "random walk attack", where the attacker randomly starts with a normal state that could not reach the unsafe state in the original FSM. Then random inputs are generated so that the attacker could move around in the completely specified circuit implementation of the FSM. When the attacker reaches the unsafe stated, we mark the state as breached. For this experiment, we attempt to breach an unsafe state 10,000 times from a random starting state, and allow the attacker to generate up to 100 random inputs. For each circuit, we choose 5 unsafe states to attack. If a circuit has less than 5 unsafe states, we test all of them. Table II shows the results of our testing. For all of the rows with "n/a", those circuits do not have any unsafe states. The last column shows a very high breaching rate, 63.28% on average and close to 100% for almost half of the circuits. The three columns in the middle indicate that among the 10,000 attempts, in the worst case the attacker can succeed with only one or two input values. And in all but 4 benchmarks, the average input length to gain unauthorized access is less than 20.

Table II Breaching the unsafe States

MCNC Circuit Index	Average number of breaches (out of 10000)	Average number of inputs	Average maximal number of inputs	Average minimal number of inputs	Average breach rate
1	913	9.68	58.50	1.50	9.13%
2	1252.6	10.29	22.80	1.60	12.53%
3	7457	2.51	17.00	1.00	74.57%
4	n/a	n/a	n/a	n/a	n/a

MCNC Circuit Index	Average number of breaches (out of 10000)	Average number of inputs	Average maximal number of inputs	Average minimal number of inputs	Average breach rate
5	9779.5	21.15	99.00	1.00	97.80%
6	10000	3.18	24.00	1.00	100.00%
7	n/a	n/a	n/a	n/a	n/a
8	10000	4.63	31.25	2.00	100.00%
9	10000	4.18	33.60	1.60	100.00%
10	8701	17.21	70.80	1.80	87.01%
11	9953.2	18.26	98.40	4.60	99.53%
12	9989.8	8.64	60.20	1.20	99.90%
13	9998	12.11	94.00	1.00	99.98%
14	9927	13.10	71.80	1.80	99.27%
15	n/a	n/a	n/a	n/a	n/a
16	9633.4	22.23	83.60	4.60	96.33%
17	5.2	6.10	23.20	3.00	0.05%
18	0	0.00	0.00	0.00	0.00%
19	n/a	n/a	n/a	n/a	n/a
20	7165.4	42.24	100.00	3.40	71.65%
21	379.2	1.49	4.80	1.20	3.79%
22	1224.8	51.72	99.20	4.60	12.25%
23	2707.67	2.29	11.67	1.33	27.08%
24	7479.5	2.02	13.50	1.00	74.80%
Average	6328.31	12.65	50.87	1.96	63.28%

A malicious designer of a sequential system has access to the original system specification and can add transitions to the blif file before FSM synthesis. (Note that we do not consider the case that the attacker removes or changes transitions from the blif file. In that case, the required functionality of the FSM will not be implemented and such an attack can be detected relatively easily by verification tools.) For an attacker to gain unauthorized access to a protected state while hiding this malicious behavior, the attacker only need to add one transition, for example by specifying a previously *don't care* transition from a normal state to the protected state to make the state unsafe. As we have discussed earlier, a naïve way to prevent such attack is to make the FSM completely specified by specifying all the *don't care* states and the *don't care* transitions.

We now report the impact on design quality by this simple attack and its naïve countermeasure. We use area, power, maximal negative slack (the difference between the circuit's timing requirement and the longest delay from input to output, which measures how good the design meets its timing requirement), and the sum of negative slack as the metrics for design quality. The original FSM is synthesized once to give us the baseline for comparison. We assume that the malicious attacker will add only one transition to breach the system. We allow the malicious attacker to add different transitions and design the system 10 times. The overhead of best malicious design and the average overhead over all the malicious designs compared with the baseline are reported as well as the design overhead when we apply the simple countermeasure to ensure trust in the design. In summary, in 15 of the 24 circuits, the average change to the malicious circuits' statistics (area, delay, and power usage) is within ±10%. **The average design overhead is 5.7% on area, 6.7% on delay, and 6.8% on power.** If the malicious designer uses the best of the 10 designs, **the quality of the malicious design indeed improves 10.0% on area, 9.6% on delay, and 11.4% on power**. Furthermore, such overhead is on the sequential component of the circuit only. If we consider the entire circuit with the combinational circuitry, the overhead will become even smaller. Therefore, it will not be effective to detect malicious design by evaluating the design quality metrics. On the other hand, it is possible to apply the naïve approach by simply creating a sink state out of an unused state and redirecting all the *don't care* transitions to this sink state. This ensures the trust in the design, at the cost of very high performance overhead, **89.6% on area, 66.4% on delay, and 92.8% on power,** which will most likely make this approach impractical,.(see [3] for the detailed report).

As explained in the previous section, if we do not care about the next state as long as it is not a safe state, we can use flip flops with controlled input to establish trust in the logic implementation of an FSM. To reduce the overhead caused by these controlled input signals, the FSM must be edited. If the number of safe states is less than a power of two, some of the safe states need to be duplicated using a modified version of the process in [29] to duplicate the states that will cause the least, or reduce, the overhead. The next step requires that we partially encode our FSM using a slightly modified version of the encoding algorithm. For example, if we have an FSM with 30 states and 8 states to be protected, all safe states will be given the same partial encoding in the format 11XXX. The rest of the states will have the encodings of 10XXX, 01XXX, or 00XXX. The state encoding algorithm or tool will then fill in the Xs with 0s or 1s in the most efficient manner. Once this process is complete, the original FSM's states are encoded using the same process but all of the states start with the partial encoding comprised of all Xs. The two FSMs are then compared and the overhead is calculated for the FSM that has been modified for protection of the safe states in comparison to the original FSM. The most important result is that our proposed approach can reduce these overhead to **7.01% on area, 2.82% on delay, and 6.33% on power,** which is a huge improvement over the naïve approach.

3. Summary of Other Publications Related to This Grant

The main topic we described in the above two sections is on the principles of designing trusted computing systems in general [1-3]. We have investigated how to enhance system's trust by leveraging both the underlying hardware of the system (silicon PUF) and the environment where the system operates (ENF signals). We also study how to improve the trust in a wireless sensor network with the presence of insider attackers. In this section, we report briefly our findings along these lines. More detail can be found in our published work [4-21].

Silicon physical unclonable function (PUF) is a promising solution for security challenges such as device authentication and cryptographic key storage: based on the intrinsic fabrication variation in delay, capacitance, or threshold voltage, a unique response to a given challenge can be produced to authenticate a device and a bitstream can be generated reliably as the cryptographic key. Rring oscillator (RO) PUF is a popular silicon PUF that can generate highly reliable unclonable outputs by amplifying the delay difference caused by fabrication variations through the substructure of ring oscillators. Our study of RO PUF focuses on its usability characterized by hardware efficiency, reliability, and applications.

[4] reports a revolutionary change in the design of PUF as we generate multiple bits of PUF secrecy simultaneously instead of "one bit at a time or by a piece of PUF circuitry". The basic idea is to compare the frequency of a group of ROs, not just one pair of ROs. This can significantly improve the hardware utilization of PUF circuitry. [5] gives a new syndrome coding scheme for this group-based PUF that facilitates error correction. [6] describes the overall design and implementation issues related to the group-based RO PUF. [7] is another revolutionary where we selectively choose the invertors to construct ring oscillators such that a pair of ROs can have large discrepancy in frequency which will improve PUF data's robustness and reliability. We observe that the PUF data does not pass NIST's standard tests for randomness, which can be a serious concern for the data's security. Therefore, we propose a regression model to filter out the systematic variation from fabrication and successfully make the data statistically random [8-10]. Meanwhile, we propose to use PUF for the protection of FPGA design intellectual properties [11,12].

Electrical network frequency (ENF) signals from the power grid exist anywhere the power grid goes. We discover that the ENF signals also carry unique features about the power grid and more interestingly, these features can be extracted from multimedia recordings [13, 14]. We study the security issues of this power grid signature, particularly on how it can be used for anti-forensics and the corresponding countermeasures [15, 16]. As one important application of the ENF signal analysis, we develop geo-location estimation algorithms to estimate the location of a device based on the ENF signals extracted from the multimedia recordings taken by the device [17,18]. With the ubiquitous of power grid and more and more recording devices (such as sensors) deployed on all kinds of systems (such as smart phones), this opens the door to another novel and powerful tool to improve the trust of computing systems.

Finally, we consider trust on a much larger scale system, the wireless sensor network, where there exist compromised sensors (just like hardware Trojan horse) who will drop the packets that they are supposed to forward. This is known as the insider packet drop attacks and the most effective mitigation methods are based on trust, which is a measure of how much a sensor believe its neighbor will forward the packet. We identify the vulnerabilities of the current trust-based methods, discover new attacks that can exploit these vulnerabilities, and propose countermeasures. [19] reveals a selective forwarding-based denial-of-service attack where the attacking node can disconnect the victim nodes from the rest of the network without being discovered. We develop techniques that not only detects, but also prevents such attacks. [20] presents a new trust metric that can dramatically shortens the time to identify packet drop attackers in the network. [21] describes an approach to detecting and recovering trusted nodes that have been misclassified as attackers.

4. Personnel Involved in the Project

- PI: Dr. Gang Qu
- Co-PI: Dr. Min Wu
- Graduated Students and their thesis titles:
  - Dr. Youngho Cho (Graduated in Summer 2013)
    Thesis: *Trust-based Defense against Insider Packet Drop Attacks in Wireless Sensor Networks*
  - Dr. Wei-Hong Chuang (Graduated in Fall 2012)
    Thesis: *Resiliency Assessment and Enhancement of Intrinsic Fingerprinting*
  - Dr. Ravi Garg (Graduate in Summer 2013)
    Thesis: *Time and Location Forensics for Multimedia*
  - Dr. Chi-En Yin (Graduate in Spring 2012)
    Thesis: *A Group-based Ring Oscillator Physical Unclonable Function*
- Current Students:
  - Mr. Carson Dunbar (Ph.D. expected Spring 2015)
  - Mr. Mingze Gao (Ph.D. expected Spring 2017)
  - Mr. Khai Lai (M.S. expected Fall 2014)

5. Educational Outcomes and List of Invited Talks

With the supports from this grant, four Ph.D. students have graduated and three other graduate students have established the foundation of their thesis research. Another unique educational accomplishment is that the PI has successfully integrated the concept of trusted integrated circuit design into undergraduate curriculum. The PI has taught, with four other faculty members, a special topics "cybersecurity lab" for senior computer science and computer engineering students in Fall

2012; he has developed a regular course "Reverse engineering and hardware security lab" for electrical and computer engineering students, which will be offered for the first time in Fall 2014 and every fall thereafter. In both courses, part of the research results from this project and other hardware related security issues are conveyed to undergraduate students and it was an eye-opening experience for most of them. The PI has also been invited to offer an online course on hardware security through coursera (https://www.coursera.org/course/hardwaresec), where the first offering will be November 2014 and will be repeated in 2015.

In addition to the conference presentations of our 21 papers related to this grant, the PI has also delivered 18 invited talks in various occasions. Among them, 10 were for academia, 4 were in government meetings, 2 for industry, and 2 in workshops open to mixed audience. The PI views this as a great vehicle to raise the global awareness of hardware security and trust. Below is the list of the basic information about these talks (title, place, and time).

1) "Building Trusted System by Information Hiding", College of Computer Science and Technology, University of Science and Technology of China, Hefei, P.R. China, June 28, 2010.
2) "Optimizing Physical Unclonable Function's Secret Extraction", Tsinghua University, Beijing, P.R. China, July 21, 2010.
3) "Role of Hardware in Security and a Case Study on PUF", CANDE Workshop, Monterey, California, November 4-6, 2010.
4) "Building Trusted Systems by Information Hiding in Programs", Army Research Office Special Workshop on Hardware Assurance, Arlington, Virginia, April 11-12, 2011.
5) "Information Hiding Based Trusted Computing System Design", Security and Information Operations Program Review, The Air Force Office of Scientific Research, Arlington, Virginia, September 22, 2011.
6) "Role of Hardware in Building Secure and Trusted Computing Systems", Department of Electrical and Computer Engineering, the George Washington University, December 8, 2011.
7) "Enhancing Cybersecurity with Trusted Hardware", *first annual MC^2 Symposium*, College Park, Maryland, May 15, 2012.
8) "Several Problems in Computer Hardware Security and Trust for Mathematicians", *Mathematics in 21st Century: Advances and Applications*, Hefei, China, July 7-9, 2012.
9) "Information Hiding Based Trusted Computing System Design", Security and Information Operations Program Review, The Air Force Office of Scientific Research, Arlington, Virginia, October 5, 2012.
10) "Building Trusted Wireless Sensor Networks: From Circuits to Routing Protocols", Department of Electrical and Computer Engineering, University of California, Davis, California, March 19, 2013. (Host: Prof. Zhi Ding)
11) "Design of Trusted and Energy Efficient Systems", Intel Corp., Santa Clara, California, March, 26, 2013. (Host: Dr. Meiyuan Zhao)

12) "Hardware in Cybersecurity: from the Weakest Link to Great Promises", Harbin Institute of Technology, Shenzhen, China, May 26, 2013. (Host: Prof. Aijiao Cui)

13) "Designing Trusted Energy-Efficient Circuits and Systems", Center for Energy-Efficient Computing and Applications, Peking University, Beijing, China, May 30, 2013. (Host: Dr. Peng Li)

14) "Information Hiding Based Trusted Computing System Design", Security and Information Operations Program Review, The Air Force Office of Scientific Research, Arlington, Virginia, August 5, 2013.

15) "Designing Trusted Energy-Efficient Circuits and Systems", School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia, September 11, 2013. (Host: Prof. Marilyn Wolf)

16) "Hardware in Cybersecurity: from the Weakest Link to Great Promises", PRECISE seminar, Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, Pennsylvania, September 18, 2013. (Host: Prof. Insup Lee)

17) "Hardware in Cybersecurity: from the Weakest Link to Great Promises", SANG Seminar, Department of Computer Science, Volgenau School of Engineering, George Mason University, Fairfax, Virginia, October 4, 2013. (Host: Prof. Songqing Chen)

18) "Hardware in Cybersecurity: from the Weakest Link to Great Promises", QualComm, San Diego, California, July 22, 2014. (Host: Dr. Eric Guo).

References (the first 21 are our publications based on work supported by this grant)

[1] C. Dunbar and G. Qu, "Designing Trusted Circuits from Finite State Machines", *17th International Workshop on Logic and Synthesis (IWLS'13)*, pp. 104-111, June 2013.

[2] C. Dunbar and G. Qu, "A Practical Circuit Fingerprinting Method Utilizing Observability Don't Care Conditions", *18th International Workshop on Logic and Synthesis (IWLS'14)*, pp. 1-8, June 2014.

[3] C. Dunbar and G. Qu, "Designing Trusted Embedded Systems from Finite State Machines", *ACM Transactions on Embedded Computing Systems (TECS)* (accepted in June 2014)

[4] C. Yin and G. Qu. "LISA: Maximizing RO PUF's Secret Extraction," *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST 2010)*, pp. 100-105, June 2010.

[5] C. Yin and G. Qu, "Kendall Syndrome Coding (KSC) for Group-based Ring-Oscillator Physical Unclonable Functions", (work-in-progress) *49th ACM/IEEE Design Automation Conference (DAC'12),* June 2012.

[6] C. Yin, G. Qu, and Q. Zhou, "Design and Implementation of a Group-based RO PUF", *Design, Automation and Test in Europe (DATE'13)*, pp. 416-421, March 2013.

[7] M. Gao, K. Lai, and G. Qu, "A Highly Flexible Ring Oscillator PUF", *51st ACM/IEEE Design Automation Conference (DAC'14)*, June 2014.

[8] C. Yin and G. Qu, "Obtaining Statistically Random Information from Silicon Physical Unclonable Functions", *IEEE Transactions on Emerging Topics in Computing (TETC)* (accepted in December 2013)

[9] C. Yin and G. Qu, "A Regression-Based Entropy Distiller for RO PUFs", (work-in-progress) *49th ACM/IEEE Design Automation Conference (DAC'12),* June 2012.

[10] C. Yin and G. Qu, “Improving PUF Security with Regression-based Distiller”, *50th ACM/IEEE Design Automation Conference (DAC’13),* pp. 1-6, June 2013.

[11] J. Zhang, Y. Lin, Y. Lyu, G. Qu, R. Cheung, W. Che, Q. Zhou, and J. Bian, “FPGA IP Protection by Binding Finite State Machine to Physical Unclonable Function”, *23nd International Conference on Field Programmable Logic and Applications (FPL’13)*, pp. 1-4, September 2013.

[12] J. Zhang, Q. Wu, Y. Lyu, Q. Zhou, Y. Cai, Y. Lin and G. Qu, “Design and Implementation of a Delay-based PUF for FPGA Intellectual Property Protection”, *13th International Conference on Computer-Aided Design and Computer Graphics*, pp. 107-114, November 2013.

[13] *R. Garg, A.L. Varna, A. Hajj-Ahmad*, and **M. Wu**: “ ‘Seeing’ ENF: Power Signature Based Timestamp for Digital Multimedia via Optical Sensing and Signal Processing,” IEEE Trans. on Info. Forensics and Security, vol. 8, no. 9, pp. 1417-1432, September 2013.

[14] *R. Garg*, *A.L. Varna* and **M. Wu**, " 'Seeing' ENF: Natural Time Stamp for Digital Video via Optical Sensing and Signal Processing," long paper accepted by ACM Multimedia, Scottsdale, Arizona, Nov. 2011. [**Best Student Paper Award**]

[15] *W-H. Chuang*, *R. Garg*, and **M. Wu**: “Anti-Forensics and Countermeasures of Electrical Network Frequency Analysis,” IEEE Trans. on Info. Forensics and Security, vol. 8, no. 12, pp.2073-2088, Dec. 2013.

[16] *W-H. Chuang*, *R. Garg*, and **M. Wu**, “How Secure are Power Network Signature Based Time Stamps?” Proc. of 19thACM Conference on Computer and Communications Security (CCS), Raleigh, NC, Oct. 2012.

[17] *R. Garg*, *A. Hajj-Ahmad* and **M. Wu**, “Geo-Location Estimation from Electrical Network Frequency Signals,” Proc. of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2013), Vancouver, Canada, May 2013.

[18] *A. Hajj-Ahmad*, *R. Garg*, and **M. Wu**: “ENF Based Location Classification of Sensor Recordings,” Proc. of IEEE Workshop on Info. Forensics and Security (WIFS), Guangzhou, China, Nov. 2013.

[19] Y. Cho and G. Qu, “Detection and Prevention of Selective Forwarding-based Denial-of-Service Attacks in WSNs,” *Hindawi International Journal of Distributed Sensor Networks (IJDSN)*, Vol. 2013, Article ID 205920, pp.1 – 16, http://dx.doi.org/10.1155/2013/205920.

[20] Y. Cho, G. Qu, and Y. Wu. “Insider Threats against Trust Mechanism with Watchdog and Defending Approaches in Wireless Sensor Networks”, IEEE Symposium on Security and Privacy Workshops, Workshop on Research for Insider Threat (WRIT’12), pp. 134-141, May, 2012.

[21] Y. Cho and G. Qu, “FADER: False Alarm DEtection and Recovery for Trust-aware Routing in Wireless Sensor Networks”, 2013 International Conference on Connected Vehicles & Expo (ICCVE 2013), December 2013.

[22] C.E. Irvine and K. Levitt. “Trusted Hardware: Can It Be Trustworthy?”, *ACM/IEEE Design Automation Conference*, pp. 1-4, June 2007.

[23] S. Trimberger. “Trusted Design in FPGAs”, *ACM/IEEE Design Automation Conference*, pp. 5-8, June 2007.

[24] G.E. Suh and S. Devadas. “Physical Unclonable Functions for Device Authentication and Secret Key Generation”, *ACM/IEEE Design Automation Conference*, pp. 9-12, June 2007.

[25] Linkages: Manufacturing Trends in Electronics Interconnection Technology, Committee on Manufacturing Trends in Printed Circuit Technology, National Research Council, 2005 http://books.nap.edu/catalog.php?record_id=11515

[26] Report of the Defense Science Board Task Force on High Performance Microchip Supply, February 2005.

[27] B.S. Cohen. "On Integrated Circuits Supply Chain Issues in a Global Commercial Market –Defense Security and Access Concerns", March 2007. http://armedservices.house.gov/pdfs/TUTCtech031407/Cohen_Testimony031407.pdf

[28] M. LaPedus. "IC Manufacturing Set for Restructuring", EE Times, November 2006. http://www.eetimes.com/news/latest/ showArticle.jhtml?articleID=193600424

[29] L. Yuan, G. Qu, T. Villa, and A. Sangiovanni-Vincentelli. "FSM Re-Engineering: A Novel Approach to Sequential Circuit Synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, Vol. 27, No. 6, pp. 1159-1164, June 2008.

[30] Berkeley Logic Interchange Format (BLIF): http://www.ece.cmu.edu/~ee760/760docs/blif.pdf